

Table based KNN for Automatic Keyword Extraction combined with Text Classification

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the table based KNN algorithm as the approaches to the text classification and the keyword extraction, in order to automate fully the latter. In the previous work, we proposed the table based KNN algorithm as an approach to the keyword extraction, but the domain should be presented as a tag, in preparing a text as the input. In this research, the KNN algorithm is also modified into the table-based version and applied to both tasks. In the proposed system, a text without its domain tag is given as the input, its domain is automatically assigned to it through the text classification, and keywords are extracted from it by the classifier which corresponds to the domain. The goal of this research is to automate the keyword extraction and improve the performance of both tasks, using the table based KNN version.

I. INTRODUCTION

The keyword extraction is defined as the process of selecting and extracting important words from a text with respect to its contents. The keyword extraction is interpreted into a binary classification, in order to apply a machine learning algorithm. The process of keyword extraction is to index a text which is given as the input into a list of words, classify each word into keyword or non-keyword, and extract words which are classified into keyword. The classification task which is mapped from the keyword extraction belongs to the domain dependent task where each entity may be classified differently depending on the given domain. It is required to present the domain to the keyword extraction system for executing it.

This research is motivated by the requirement of a domain as the input as well as a text for doing the keyword extraction. The results from applying the table-based version of the KNN algorithm to the keyword extraction were successful in the previous work. Because the classification which is mapped from the keyword extraction is a domain dependent classification, the domain is required for nominating the classifier which corresponds to it. It is possible to assign the domain automatically to a text through the text categorization. The problem is solved by connecting the keyword extraction with the text classification.

The idea of this research is to apply the table based KNN algorithm to the keyword extraction which relates to

the text categorization. The similarity metric between tables is defined, and the KNN algorithm is modified into the table-based version. It is applied to the text categorization which assigns a domain automatically to the input text. An input text is indexed into a list of words, each of them is classified into keyword or non-keyword by the modified KNN algorithm, and ones which are classified into keyword are extracted as the output. The table based KNN algorithm is adopted for implementing both the text categorization and the keyword extraction.

Let us mention some benefits which are provided by this research. The problems in encoding words or texts into numerical vectors are avoided by encoding them into tables as alternative representations. The performance of the keyword extraction and the text classification is improved by adopting the table based KNN version. We do not need to present the domain as a tag of the input text by connecting the keyword extraction with the text classification. In this research, we upgrade the keyword extraction system by automating the process of nominating a suitable classifier.

Let us mention remaining tasks as the further research. Texts and words may be encoded into compound representations each of which consists of more than one structured form. Other operations on tables may be defined as well as the similarity metric. Other machine learning algorithms and deep learning algorithms may be modified into table-based versions. The keyword extraction system may be implemented as a real program and a library module by adopting the proposed approach.

II. PROPOSED APPROACH

A. Similarity Metric

This section is concerned with the table as the representation of a word. The table is defined as a set of entries, each of which consists of an element and its weight. The table which represents a word consists of entries each of which contains a text identifier and its weight. The similarity between tables is computed based on shared words as the semantic similarity between words. This section is intended to describe the process of encoding a word into a table, and the process of computing the similarity between tables.

The process of encoding words into tables is illustrated in Figure 1. In the inverted indexing, each word is linked with texts which include itself. In the text-word weighting, its weights in the lined texts are computed; a weight indicates a relationship between a word and a text. Entries with their lower weights are removed for downsizing the table. Refer to [1] for studying the encoding process in more detail.

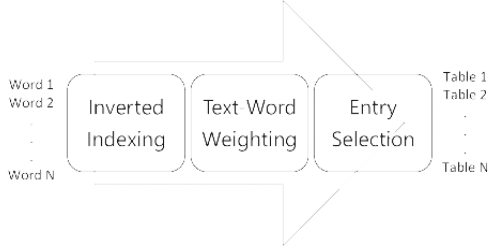


Figure 1. Process of Encoding Words into Tables

Let us mention the function of a table which maps it into a set of text identifiers as shown in Figure 2. The table which represents a word is expressed as entry set as shown in equation (1),

$$T = \{(d_1, w_1), (d_2, w_2), \dots, (d_{|T|}, w_{|T|})\} \quad (1)$$

where d_i is a text identifier and w_i is the weight of the word, T , in the text, d_i . The function for generating a list of text identifiers is expressed as equation (2),

$$F(T) = \{d_1, d_2, \dots, d_{|T|}\} \quad (2)$$

The function is the operation which takes only text identifiers without their weights as a set. The operation is applied to computing the similarity between tables which represent words.

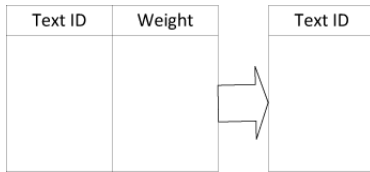


Figure 2. Conversion of Table into Text Set

Let us mention the process of computing the similarity between two tables as a semantic similarity between words. The two tables which represent words are notated respectively by $T_1 = \{(d_{11}, w_{11}), (d_{12}, w_{12}), \dots, (d_{1|T_1|}, w_{1|T_1|})\}$ and $T_2 = \{(d_{21}, w_{21}), (d_{22}, w_{22}), \dots, (d_{2|T_2|}, w_{2|T_2|})\}$. The intersection between the two sets which is generated by applying the function, $F(\cdot)$ as shown in equation (3),

$$F(T_1) \cap F(T_2) = \{d_{s1}, d_{s2}, \dots, d_{s|F(T_1) \cap F(T_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared text identifier, d_{si} , its weight from the table, T_1 , w_{si}^1 , and

its weight from the table, T_2 , w_{si}^2 , is expressed as equation (4),

$$ST_{12} = \{(d_{s1}, w_{s1}^1, w_{s1}^2), (d_{s2}, w_{s2}^1, w_{s2}^2), \dots, (d_{s|F(T_1) \cap F(T_2)|}, w_{s|F(T_1) \cap F(T_2)|}^1, w_{s|F(T_1) \cap F(T_2)|}^2)\} \quad (4)$$

The similarity between tables, T_1 and T_2 is computed by equation (5),

$$sim(T_1, T_2) = \frac{2 \left(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^2 \right)}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_2|} w_{2i}}. \quad (5)$$

The value which is computed from equation (5) is always given as a normalized value between zero and one as shown in equation (6),

$$0 \leq sim(T_1, T_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of similarity between words. A word is encoded into table by the inverted indexing, the text identifier weighting, and the entry selection. A table is expressed as a set of entries and filtered into a set of text identifiers by the function. The similarity between tables is computed based on their shared entries by equation (5). In the future research, we consider representing a word into multiple tables.

B. Table based KNN Algorithm

This section is concerned with the table based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [22]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into tables, and they are notated by a set $Tr = \{T_1, T_2, \dots, T_N\}$. A novice word is encoded into a table notated by T . Its similarities with the tables which represent the sample words are computed by equation (5), as $sim(T, T_1), sim(T, T_2), \dots, sim(T, T_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The

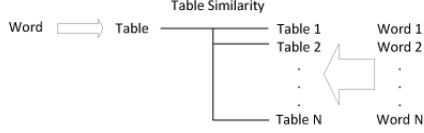


Figure 3. Similarities of Novice Input with Training Examples

training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (7),

$$Nr = \text{Select}_k(Tr, T), Nr \subseteq Tr \quad (7)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (8),

$$Nr = \{T_1^{near}, T_2^{near}, \dots, T_k^{near}\} \quad (8)$$

The elements in the set, Nr , are used for voting their labels in the next step.

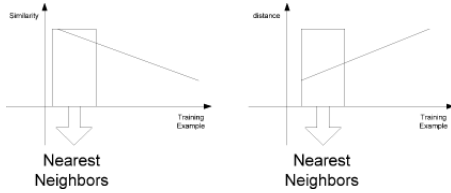


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(T_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, T , is decided by equation (9),

$$\text{label}(T) = \arg\max_{j=1}^m \text{Count}(Nr, c_j) \quad (9)$$

The process of deciding the label of a novice item by equation (9), is called voting.

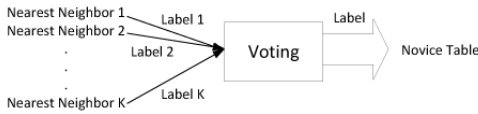


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the table-based version of KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided

by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the keyword extraction system which adopts the table based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the keyword extraction. It is viewed as a binary classification which classifies a word into keyword or non-keyword. This section is intended to describe the keyword extraction system with respect to its function and its architecture.

The process of collecting the sample words and encoding them into tables for implementing the keyword extraction system. The keyword extraction is viewed as a binary classification which classifies a word into keyword or non-keyword is illustrated in Figure 6. The topic-based word classification belongs to the domain independent classification where a same word is always classified identically with regardless of the domain, whereas the keyword extraction is the domain dependent classification where a same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is required to tag the input text with its own domain for executing the keyword extraction.

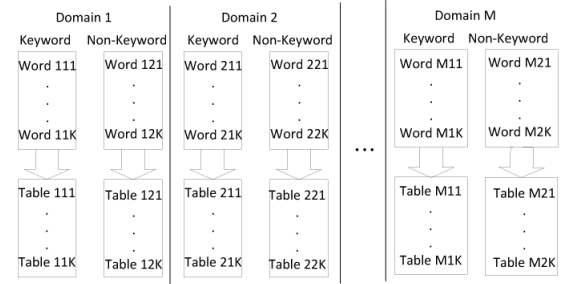


Figure 6. Preparation of Training Examples

The entire architecture of the proposed keyword extraction system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into tables. The words which are indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: indexing module, encoding module, similarity computation module, and voting module.

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which are labeled with keyword or non-keyword are collected

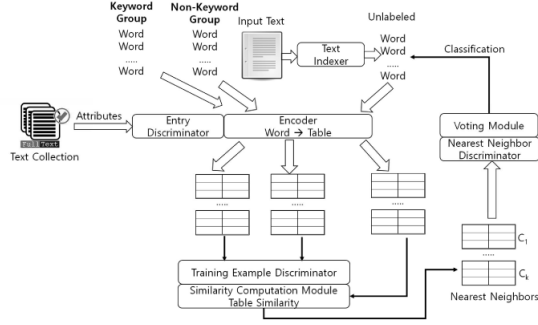


Figure 7. System Architecture

within a domain, and they are encoded into tables. An input text is indexed into a list of words, and they are also encoded into tables. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. The words which are classified into keyword are extracted as the final output.

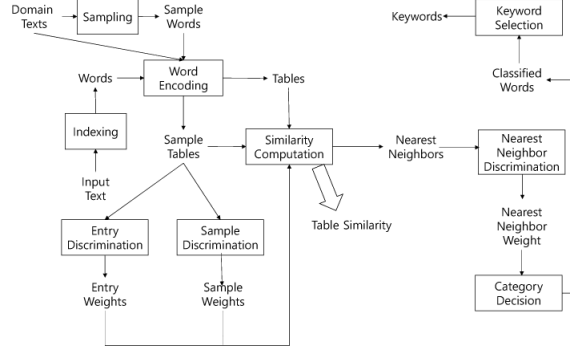


Figure 8. System Flow

Let us make some remarks on the proposed system whose architecture is presented in Figure 7. The keyword extraction is defined as a binary classification where each word is classified into keyword or non-keyword. Each word is encoded into a table instead of a numerical vector and is classified directly. Among words which are included in the input text, ones which are classified into keyword are selected. We need to add the text classification module for predicting the domain of the input text automatically.

D. Connection with Text Classification

This section is concerned with the connection of the keyword extraction with the text classification. In the previous section, we mentioned the keyword extraction system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section II-B is used for implementing the text classification. This section is intended

to describe the connection of the keyword extraction system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into tables by the process which is described in Section II-A. The similarity computation module is for computing the similarities between tables which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system used for automating deciding the domain of the input text.

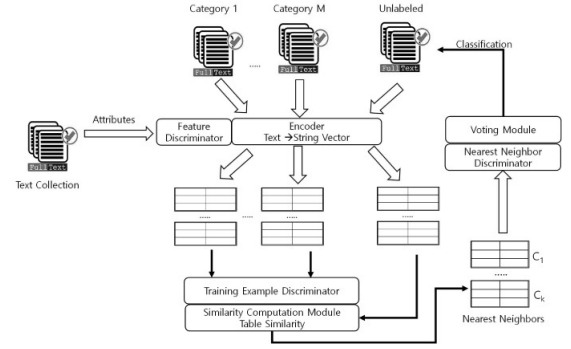


Figure 9. Text Categorization System

The connection of the keyword extraction with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the keyword extraction system which is described in Section II-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the keyword extraction, automatically.

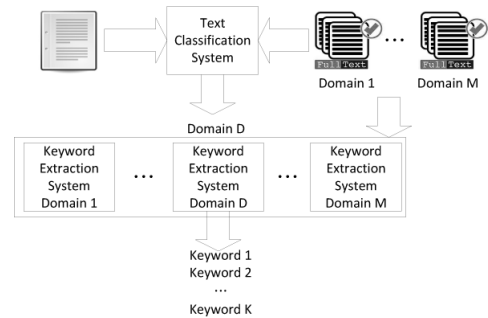


Figure 10. Keyword Extraction System connected with Text categorization

The process of executing the keyword extraction, connecting with the text classification is illustrated in Figure 11. Sample texts each of which is labeled with its own domain are prepared, and sample words each of which is labeled with keyword or non-keyword are prepared in each domain. A novice text is classified into one among the predefined domains, and the classifier which corresponds to the domain is nominated. The novice text is indexed into a list of words, and each word is classified into keyword or non-keyword. The list of words which is classified into keyword is the final output from this system; the domain which is assigned to the input text is the guide for selecting a classifier.

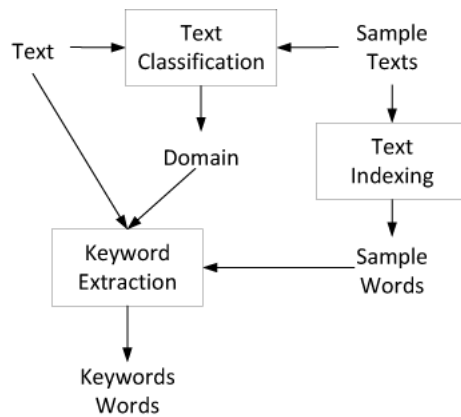


Figure 11. System Flow of Keyword Extraction connected with Text Categorization

Let us make some remarks on the connection of the keyword extraction with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the keyword extraction is to extract important words from a text. In the execution flow, the input text is classified into a domain, it is indexed into a list of words, and each word is classified into keyword or non-keyword. We consider connecting the text classification as the main task the keyword extraction as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Table based K Nearest Neighbor for Word Categorization in News Articles", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.
- [2] T. Jo, "Keyword Extraction in News Articles using Table based K Nearest Neighbors", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.